ChinaXiv合作期刊

# 格上具有完全前向安全的 0 轮往返时间密钥交换协议 *

赵宗渠，马少提，汤永利，叶　青†
(河南理工大学 计算机科学与技术学院, 河南 焦作 454000)

**摘　要：**0-RTT 密钥交换协议，允许客户端在零往返时间发送加密保护的有效载荷和第一条密钥交换协议消息，具有非交互、可离线等优点。为了降低密钥交换往返时间，基于穿透加密思想提出一种格上 0-RTT 密钥交换协议，首先利用一次性签名算法和分级身份基密钥封装机制构造可穿透前向保密密钥封装方案，然后使用可穿透前向保密密钥封装方案设计 0-RTT 密钥交换协议。协议只需客户端对服务器进行单向认证，并且能够有效抵抗量子攻击和重放攻击。与同类协议相比，所提协议具有可穿透的完全前向安全，减少了通信轮数，提高了通信效率。
**关键词：**格；密钥交换；0-RTT；前向安全
**中图分类号：**TP309　　doi: 10.19734/j.issn.1001-3695.2020.02.0072

## Zero round trip time key exchange protocol with full forward secrecy on lattice

Zhao Zongqu, Ma Shaoti, Tang Yongli, Ye Qing†
(*College of Computer Science & Technology*, *Henan Polytechnic University*, *Jiaozuo Henan* 454000, *China*)

**Abstract:** 0-RTT key exchange protocol allows clients to send encrypted protected payloads and the first key exchange protocol message at zero round-trip time, which has the advantages of non-interactive and off-line. In order to reduce the round-trip time of key exchange, this paper proposed a 0-RTT key exchange protocol on lattice based on the idea of penetrating encryption. Firstly, it utilized the one-time signatures algorithm and the hierarchical identity-based key encapsulation mechanism to construct penetrable forward secret key encapsulation scheme, and then used the penetrable forward secret key encapsulation scheme to design a 0-RTT key exchange protocol. The protocol only required the client side to authenticate the server one-way, and could effectively resist the quantum attack and replay attack. Compared with similar protocols, the proposed protocol has penetrable full forward secrecy, reduces the number of communication rounds and improves the communication efficiency.
**Key words:** lattice; key exchange; zero round-trip time(0-RTT); forward secrecy

chinaXiv:202009.00074v1

## 0　引言

认证密钥交换(authenticated key exchange，AKE)协议是实体进行安全通信的前提，允许通信双方在公开的信道上建立一个共享的高熵会话密钥,并用这个会话密钥进行加密消息、认证和完整性校验等工作。1976 年，Diffie 和 Hellman[1]基于离散对数困难问题设计了第一个密钥交换协议，简称为“DH 协议”，这一开创性的研究成果加速了密钥交换协议领域的发展。由于 DH 协议是被动安全的密钥交换协议，无法抵抗主动攻击，存在会话密钥泄露的风险，因此许多基于 DH 假设[2~4]的密钥交换协议通过交换密钥的部分元素或加入身份信息来抵御主动攻击。

像 TLS 这样的经典 AKE 协议在传输第一个有效的实际数据消息之前需要交换大量协议信息协商共享的会话密钥，因此会产生相当大的延迟开销。延迟通常是以往返时间(RTT)来衡量的，在发送第一个实际数据之前，必须进行 N 轮往返消息传递，即 N-RTT。传输层安全( transport layer security,TLS)协议提供一个认证密钥交换，允许两个远程方通过不安全的通道建立共享的会话密钥。TLS1.2[5]协议需要两次往返时长(2-RTT)完成握手，然后才能发送请求；TLS1.3[6]协议比 TLS1.2 更快更安全，TLS 握手仅需要一次往返时长(1-RTT)，如果网站以前被客户端连接过，则 TLS 握手的往返时长为零。作为高性能 AKE 协议的 HMQV[7]在协商会话密钥时也需要至少发送两条消息(即 1-RTT)。

将密钥交换协议的延迟开销降低到零往返时间(0-RTT)，同时保持严格的安全保证已成为学术界和工业界的一个主要设计目标。从实用的角度来看，谷歌的 QUIC 协议[8]不仅把延迟开销降到了零往返时间，而且已经在 Google Chrome 和 Opera Web 浏览器中得到应用，并在 2015 年由 Google 向 IETF 提议作为 IETF 标准。

2017 年，Günther, Hale, Jager 和 Lauer[GHJL17][9]提出了基于穿透加密且具有完全前向安全的 0-RTT 密钥交换协议，该协议是在 Canetti,Halevi 和 Katz[10]的前向安全公钥加密和 Green 等[11]的前向保密穿透公钥加密的工作基础上构建得到的。GHJL17[9]方案通过一次性签名技术和 Blazy 等[12]的身份基分级密钥封装机制构建的可穿透前向保密密钥封装机制来实现前向保密一次通过密钥交换协议，该方案使前向保密 0 轮往返时长密钥交换协议成为了可能，并且具有完全前向安全，能够抵抗重放攻击。2018 年，Derler 和 Jager 等[13]使用 Bloom Filter Encryption(BFE)构造了 0-RTT 密钥交换协议，提高了计算效率，并通过容忍一个不可忽略的正确性误差把密钥的增长限制在一个可以容忍的限度内。

**收稿日期：**2020-02-16；**修回日期：**2020-04-11　**基金项目：**国家自然科学基金资助项目(61802117)；河南省高新科技创新团队支持计划资助项目(20IRTSTHN013)；“河南省网络密码技术重点实验室”开放课题(LNCT2019-A04)；河南省重点研发与推广专项(科技攻关)项目(192102210280)；河南省高等学校重点科研项目(19A520025)

**作者简介：**赵宗渠(1974-)，男，河南沁阳人，讲师，博士，主要研究方向为密码学、网络安全、恶意代码分析；马少提(1995-)，男，河南许昌人，硕士研究生，主要研究方向为信息安全、密码学；汤永利(1972-)，男，河南孟州人，教授，硕导，博士，主要研究方向为信息安全、密码学；叶青(1981-)，女(通信作者)，辽宁营口人，讲师，主要研究方向为信息安全、密码学(yeqing@hpu.edu.cn).

ChinaXiv合作期刊


随着量子计算理论的发展，基于传统大整数分解和离散对数困难问题的 AKE 协议[14]并不能抵抗量子攻击，在后量子时代，这些方案所依赖的困难问题已经可以用量子算法在多项式时间内解决，而基于格上困难问题的公钥密码方案在量子理论下还不存在多项式时间高效求解算法，同时格上的运算是矩阵向量上的乘法，具备并行计算、效率高的特点。

基于格的密码学受到广泛关注，2009 年 Katz 等[15]设计了一个基于格的 CCA(chosen-ciphertext attack)安全的加密体制，并通过该加密体制构造近似平滑投射哈希( approximate smooth projection hash,ASPH)函数，提出第一个基于格的 2PAKE(two-party password-based anthenticated key exchange)协议。2011 年，Ding 等[16]在 Groce-Katz 框架[17]的基础上结合 Katz 等[15]提出的加密体制和近似平滑投射哈希函数，提出了一种基于格上困难问题的高效 PAKE 协议，并且在标准模型下证明了协议的安全性。2017 年，Zhang 等[18]将拆分的公钥加密体制和 Katz 等[15]的 3 次通信框架相结合，提出了基于格上困难问题且仅需两轮通信的 PAKE 协议，提高了通信效率，但是方案中存在拆分公钥加密造成计算花销增加的不足。2019 年，李子臣等[19]基于环上误差学习问题设计了一种后量子认证密钥交换协议，在标准 eCK 模型下可证明安全并达到弱的完全前向安全。在这些格上 AKE 协议中，协商会话密钥需要 2 轮或者 3 轮通信，所需的通信开销更多。

本文基于 GHJL17 方案[9]的设计思想构造了一种新的 0-RTT 密钥交换协议。方案的主要贡献有：1)设计了一种可穿透前向保密密钥封装机制实现穿透加密和密钥更新功能，使本文协议具有完全前向安全性；2)基于可穿透前向保密密钥封装机制构造本文协议，减少了通信轮数，降低了通信开销。本文所提协议具有完全的前向安全性和抵抗量子攻击、重放攻击的特点，并且实现了 0 轮通信，有更高的通信效率。

## 1　背景知识

**定义 1**　0-RTT 密钥交换。以 Diffie-Hellman 密钥交换为例，首先用户从之前的密钥交换中获得服务器的共享信息 $g^s$，然后选择指数 $x$ 生成密钥 $k_1 = g^{xs}$，给服务器发送用密钥 $k_1$ 加密过的数据和 $g^x$，服务器收到后选择 $y$ 生成密钥 $k_2 = g^{xy}$，发送用 $k_2$ 加密过的数据和 $g^y$ 给用户，在本次通信中双方使用 $k_2$ 作为会话密钥，在密钥协商时同时发送了加密信息和密钥交换信息，称之为 0-RTT 密钥交换。

**定义 2**　穿透加密。在一个可穿透前向安全密钥封装方案中，假设一个服务器拥有长期私钥 $sk$，当收到一条密文消息 $c_1$ 时，其中消息 $c_1$ 中封装了一个会话密钥，服务器用 $sk$ 解密消息 $c_1$ 并派生出一个新的私钥 $sk_{\backslash c_1}$，新的私钥在 $c_1$ 时穿透的，并且用于解密除了 $c_1$ 之外的所有密文，最终服务器删除 $sk$。

### 1.1　格的相关知识

**定义 3**　给定 $m$ 个线性无关的向量 $B = (b_1, \cdots, b_m) \in \mathbb{R}^{m \times m}$，格 $\Lambda \subset \mathbb{R}^m$ 定义为所有这些向量的整系数线性组合，$\Lambda = L(B) = \{\sum_{i=1}^{m} x_i b_i : x_i \in \mathbb{Z}\}$。

**定义 4**　q 元格。对于整数 $q$，满足 $q\mathbb{Z}^m \subseteq \mathbb{Z}^m$。对 $q, m, n \in \mathbb{Z}$，给定矩阵 $A \in \mathbb{Z}_q^{n \times m}$，定义：$\Lambda_q(A) = \{y \in \mathbb{Z}^m : y = A^T s (\bmod q)\}$；$\Lambda_q^{\perp}(A) = \{y \in \mathbb{Z}^m : Ay = 0 (\bmod q)\}$。

**定义 5**　对于任意 $s > 0$，以向量 $c \in \mathbb{R}^m$ 为中心，$x \in \Lambda$，参数为 $s$，在格 $\Lambda \subseteq \mathbb{Z}^m$ 上的高斯分布函数定义为 $\rho_{s,c}(x) = \exp(-(\pi \|x - c\|^2)/s^2)$。

**定义 6**　令 $\rho_{s,c}(\Lambda) = \sum_{x \in \Lambda} \rho_{s,c}(x)$，对于任意 $s > 0$，以向量 $c \in \mathbb{R}^m$ 为中心，参数为 $s$ 的格 $\Lambda$ 上的离散高斯分布定义为 $D_{\Lambda,s,c}(y) = \rho_{s,c}(y)/\rho_{s,c}(\Lambda)$。其中，$y \in \Lambda$。(若没有明确说明，$c$ 默认为 0)。

### 1.2　安全模型

在 AKE 协议可证明安全保证的研究方面，Bellare 和 Rogaway[20]两人作出了开创性的工作。文献[20]提供了适用于分布式环境的实体认证和认证密钥分发的第一种形式化处理并详细讨论了对称、双方设置下的相互认证和认证密钥交换问题，而且对于每一种情况都给出了协议满足其目标的一个定义、协议和证明，且假设存在一个伪随机函数。

本文采用 GHJL17 方案[9]中的协议安全性分析模型，用 $I = C \dot{\cup} S$ 表示系统中客户端(C)和服务器(S)建模的身份集，每个身份 $u \in I$ 与一个公钥/私钥对 $(pk_u, sk_u)$ 相关联。其中公钥部分 $pk_u$ 是一次性生成并固定的，而随着时间的推移 $sk_u$ 可以被相关参与者的会话修改。此外，每个身份 $u$ 在由 $\tau_u \in \mathbb{N}$ 表示的变量中保存局部的当前时间并初始化为 $\tau_u \leftarrow 1$。

在安全模型中，敌手 $A$ 与多个身份会话进行交互并运行前向保密一次通过密钥交换协议。$\pi_u^i$ 表示身份 $u$ 的第 $i$ 个会话，并与下述每个会话内部状态变量有关联：

a) $role \in \{client, server\}$ 表示会话的角色。分别要求 $role = client$ 和 $role = server$ 当且仅当 $u \in C$ 和 $u \in S$。

b) $id \in I$ 表示会话的拥有者(例如 $u$ 拥有 $\pi_u^i$)。

c) $pid \in I \cup \{\perp\}$ 表示预期的通信方并被精确设置一次。如果 $role = server$ 设置 $pid = \perp$ 表示客户端未通过身份验证。最初，如果 $role = server$ 还可以设置 $pid = \perp$ 表示在协议中需要学习的客户端身份。

d) $trans \in \{0,1\}^* \cup \{\perp\}$ 分别记录单一发送和接受的消息。

e) $time \in \mathbb{N}$ 记录当分别处理发送和接受消息时的时间间隔。

f) $key \in \{0,1\}^* \cup \{\perp\}$ 是会话中派生的会话密钥，用 $\pi_u^i.key$ 表示引用的特定会话状态变量。

g) $keystate \in \{fresh, revealed\}$ 表示会话密钥是否已被泄露，初始化 $keystate = fresh$。

**定义 7**　匹配会话。两个会话 $\pi_u^i$ 和 $\pi_v^j$ 如果是匹配关系则需要满足以下条件：

a) $\pi_u^i.trans = \pi_v^j.trans$ 表示两个会话共享同一个传输；

b) $\pi_u^i.time = \pi_v^j.time$ 表示两个会话在同一个时间间隔内运行；

c) $\pi_u^i.role = client \wedge \pi_v^j.role = server$ 表示两个会话以相反的角色运行；

d) $\pi_u^i.pid = \pi_v^j.id$ 表示服务器会话由客户端的预期匹配伙伴拥有；

e) $\pi_u^i.id = \pi_v^j.pid \vee \pi_v^j.pid = \perp$ 表示客户端会话属于服务器的预期伙伴，或者服务器认为其伙伴未经身份验证。

假设敌手 $A$ 控制网络，负责传输消息，从而允许任意修改、删除或重新排序消息。它可以通过以下查询与密钥交换协议和会话交互。

$NewSession(u, role, pid, m)$：初始化新会话身份 $u \in I$，角色 $role \in \{client, server\}$ 和预期通信伙伴 $pid \in I \cup \{\perp\}$ (其中服务器会话 $pid = \perp$ 表示未经身份验证的客户端伙伴)。如果 $role \neq server$，设置 $m = \perp$。

如果 $role = server$，调用 $(sk_u, k, m) \leftarrow FSOPKE.RunC(sk_u, pk_{pid})$，否则调用 $(sk_u, k) \leftarrow FSOPKE.RunS(sk_u, pk_{pid}, m)$，其中 $pk_{\perp} = \perp$。

注册一个新会话 $\pi_u^i$，其中 $\mathrm{role} = role$，$id = u$，$\mathrm{pid} = pid$，$trans = m$，$time = \tau_u$，$key = k$。

当 $role = client$ 时，返回 $m$。当 $role = server$ 时，如果 $k = \perp$ 返回 $\perp$，否则返回 $T$。

$Reveal(\pi_u^i)$：如果会话密钥是派生的，就显示特定会话的会话密钥。如果 $\pi_u^i.key \neq \perp$，设置 $\pi_u^i.keystate \leftarrow revealed$ 并返回密钥，否则返回 $\perp$。

$Corrupt(u)$：破坏身份 $u \in I$ 的长期状态。此查询最多可在每个身份 $u$ 中查询一次，并且以后不允许对会话 $u$ 进行进一

chinaXiv:202009.00074v1

ChinaXiv合作期刊



步查询。设 $Corrupt(u)$ 为 $A$ 发出的第 $\varsigma$ 次查询；设置 $\varsigma_u^{corr} \leftarrow \varsigma$，其中 $\varsigma_u^{corr} = \infty$ 表示未损坏的身份。如果出现破坏则记录身份的当前时间 $\tau_u$，并设置 $\tau_u^{corr} \leftarrow \tau_u$，最后返回 $sk_u$。

$Tick(u)$：通过调用 $sk_u \leftarrow FSOPKE.TimeStep(sk_u)$，将一些身份 $u \in I$ 的状态一次性转发。将新时间记录为 $\tau_u \leftarrow \tau_u + 1$。

$Test(\pi_u^i)$：允许敌手挑战派生的会话密钥，并且只被询问一次。该预言在安全游戏中随机选择了一个秘密比特 $b_{test} \in \{0,1\}$。如果 $\pi_u^i.key = \perp$，返回 $\perp$。设置 $\tau^t \leftarrow \pi^t.time$，如果 $b_{test} = 0$，返回 $\pi_u^i.key$，否则返回一个根据协议特定的概率优势随机选择的密钥。

## 2　格上具有前向安全的 0-RTT 密钥交换协议

为了构建本文协议，首先使用 MP12 陷门[21]实现了一个格上标准模型下的基于 SIS 困难问题的一次性签名技术(one-time signatures, OTSIG)，接着设计了一个格上标准模型下的身份基分级密钥封装机制(hierarchical identity-based key encapsulation mechanism, HIBKEM)，然后以上述两个方案为基础模块构造了可穿透前向保密密钥封装机制(puncturable forward-secret key encapsulation mechanism, PFSKEM)，最后以可穿透保密密钥封装机制设计格上具有前向安全的 0-RTT 密钥交换协议，协议具体构造如下所示。

### 2.1　一次性签名技术

一个一次性签名技术 OTSIG 由三个概率多项式时间算法组成(OTSIG.KGen, OTSIG.Sign, OTSIG.Vfy)。

OTSIG.KGen($1^n$)：算法输入安全参数 $n$，从分布 $D$ 中选择 $\bar{A} \leftarrow \mathbb{Z}_q^{n\times\bar{m}}$，$R \in \mathbb{Z}^{\bar{m}\times nk}$，让 $A = [\bar{A} \mid G - \bar{A}R]$。选择 $A_i \leftarrow \mathbb{Z}_q^{n\times nk}$，其中 $i = 0,1,\ldots,\ell$，再选择一个向量 $u \leftarrow \mathbb{Z}_q^n$。输出公钥 $pk_{OT} = (A, A_0, \ldots, A_\ell, u)$，私钥 $sk_{OT} = R$。

OTSIG.Sign($sk_{OT}$，$\mu \in \{0,1\}^\ell$)：算法让 $A_\mu = [A \mid A_0 + \sum_{i\in[\ell]} \mu_i A_i] \in \mathbb{Z}_q^{n\times m}$，其中 $\mu_i \in \{0,1\}$ 是 $\mu$ 的第 $i$ 个比特，可视为一个整数。输出 $v \in \mathbb{Z}^m$，向量 $v$ 是从 $D_{\Lambda_u^\perp}(A_\mu), s$ 中使用包含矩阵 $A$ 的陷门 $R$ 的算法 SampleR 采样得到(它也是 $A$ 的拓展 $A_\mu$ 的一个陷门)。

OTSIG.Vfy($pk_{OT}$，$\mu$，$v$)：算法让 $A_\mu = [A \mid A_0 + \sum_{i\in[\ell]} \mu_i A_i] \in \mathbb{Z}_q^{n\times m}$。如果 $\|v\| \le s\cdot\sqrt{m}$，$A_\mu \cdot v = u$ 就接受，否则拒绝。

### 2.2　分级身份基密钥封装

一个分级身份基密钥封装机制 HIBKEM 由四个概率多项式算法构成(HIBKEM.KGen, HIBKEM.Del, HIBKEM.Encap, HIBKEM.Decap)。

HIBKEM.KGen($1^n$，$1^d$)：输入一个安全参数 $n$ 和最大分级深度 $d$，调用 $\mathrm{Trap}Gen(\bar{A}, H)$ 算法生成一个均匀随机矩阵 $A = [\bar{A} \mid HG - \bar{A}T_A] \in \mathbb{Z}_q^{n\times m}$ 和 $A$ 的陷门矩阵 $T_A = [a_1 \mid a_2 \mid \cdots \mid a_\omega] \in \mathbb{Z}^{\bar{m}\times\omega}$，选取 $n$ 维均匀随机向量 $u \in R_q^n$，运行 $SampleR(1^m)$ 算法，输出 $2d$ 个矩阵 $R_{1,0}, R_{1,1}, R_{2,0}, R_{2,1}, \cdots, R_{d,0}, R_{d,1} \in \mathbb{Z}^{m\times m}$。输出主公钥 $MPK = (A, u, R_{1,0}, R_{1,1}, R_{2,0}, R_{2,1}, \cdots, R_{d,0}, R_{d,1})$ 和主私钥 $MSK = (T_A)$。假设 $MPK$ 隐式地定义身份空间 $ID$ 和密钥空间 **K**。

HIBKEM.Del($MPK, SK_{id_{|\ell}}, id$)：输入主公钥 $MPK$，$SK_{id_{|\ell}}$ 表示分级深度为 $\ell$ 时用户公钥矩阵 $A_{id_{|\ell}}$ 所对应的陷门矩阵，其中 $A_{id_{|\ell}} = A_0(R_{1,id_1})^{-1}(R_{2,id_2})^{-1}\cdots(R_{\ell,id_\ell})^{-1} \in \mathbb{Z}_q^{n\times m}$，父用户身份 $id_{|\ell} = \{0,1\}^{\ell\le d}$；输入子用户身份 $id = (id_1 \mid id_2 \mid \cdots \mid id_\ell \mid id_{\ell+1} \mid \cdots \mid id_{\bar{k}})$，其中 $\bar{k} \le d$。令 $R = (R_{\ell+1,id_{\ell+1}})^{-1}(R_{\ell+2,id_{\ell+2}})^{-1}\cdots(R_{\bar{k},id_{\bar{k}}})^{-1} \in \mathbb{Z}^{m\times m}$，$A_{id} = A_{id_\ell} R \in \mathbb{Z}_q^{n\times m}$。调用陷门派生算法得到 $S' = TrapDel(A_{id_{|\ell}}, R, SK_{id_{|\ell}}, \sigma_k)$，输出陷门矩阵 $SK_{id} = S'$。

HIBKEM.Encap($MPK, id$)：输入主公钥 $MPK$，分级深度为 $\bar{k}$ 的接收方用户身份 $id$。计算可逆矩阵 $R_{id} = (R_{1,id_1})(R_{2,id_2})\cdots(R_{\bar{k},id_{\bar{k}}}) \in \mathbb{Z}^{m\times m}$ 和用户公钥矩阵 $A_{id} \leftarrow A_0 R_{id}^{-1} \in \mathbb{Z}_q^{n\times m}$。随机选取密钥 $k \in \{0,1\}$，使用对偶 Regev 算法来加密密钥 $k$：首先选取均匀随机向量 $s \leftarrow R_q^n$；然后选取容错值 $x \xleftarrow{\bar{\Psi}_{\alpha_{\bar{k}}}} \mathbb{Z}_q$ 和容错向量 $y \xleftarrow{\bar{\Psi}_{\alpha_{\bar{k}}}^m} \mathbb{Z}_q^m$；然后计算加密密文 $CT = (c_0 = u^T s + x + k\lfloor q/2 \rfloor, c_1 = A_{id}^T s + y) \in \mathbb{Z}_q \times \mathbb{Z}_q^m$，输出密文和密钥 $(CT, K = k)$。

HIBKEM.Decap($MPK, SK_{id}, CT$)：输入主公钥 $MPK$，陷门矩阵 $SK_{id}$ 和密文 $CT$，其中用户身份 $id$ 的分级深度为 $|id| = \bar{k}$。令高斯参数 $\tau_{\bar{k}} = \sigma_{\bar{k}}\sqrt{m} \times \omega(\sqrt{\log m})$，做和封装算法一样的操作得到用户公钥矩阵 $A_{id} \in \mathbb{Z}_q^m$，运行原象采样算法 $e_{id} \leftarrow MP12\mathrm{Sample}(\mathrm{A}_{id}, \mathrm{SK}_{id}, \mathrm{u}, \tau_{\bar{k}})$，满足 $A_{id} e_{id} = u$，计算 $k' = c_0 - e_{id}^T c_1 \in \mathbb{Z}_q$，其中把 $k'$ 和 $\lfloor q/2 \rfloor$ 视为 $\mathbb{Z}$ 中的整数，如果 $|k' - \lfloor q/2 \rfloor| < \lfloor q/4 \rfloor$，输出 1，否则输出 0。

HIBKEM 正确性：HIBKEM 解封装正确性由定理 1 刻画。

**定理 1**　HIBKEM 的解封装是正确的，对任意的 $id \in ID$，$(MPK, MSK) \leftarrow$ HIBKEM. KGen$(1^n, 1^d)$，$SK_{id} \leftarrow$ HIBKEM. Del$(MPK, SK_{id_{|\ell}}, id)$ 和密钥 $k \in \{0,1\}$，其中 $ID$ 为身份空间，有 $\Pr[\mathrm{Decap}(MPK, SK_{id}, \mathrm{Encap}(MPK, id, k)) = k] = 1 - negl(n)$ 成立。

**证明**　HIBKEM 解封装算法的输出为 $k' = c_0 - e_{id}^T c_1 = u^T s + x + k\lfloor q/2 \rfloor - e_{id}^T(A_{id}^T s + y) = u^T s + x + k\lfloor q/2 \rfloor - (A_{id} e_{id})^T s - e_{id}^T y = k\lfloor q/2 \rfloor + x - e_{id}^T y$，其中 $x - e_{id}^T y$ 为误差项，其绝对值小于 $q/5$，满足 $|k' - \lfloor q/2 \rfloor| < \lfloor q/4 \rfloor$ 输出 1，否则输出 0 的设置，定理 1 成立。

HIBKEM 安全性：在可选则 ID 的游戏 $G_{A,HIBKEM}^{IND-sID-CPA}(n)$ 中把敌手 $A$ 的优势定义为 $Adv_{A,HIBKEM}^{IND-sID-CPA}(n) := \left|\Pr[G_{A,HIBKEM}^{IND-sID-CPA}(n) = 1] - \frac{1}{2}\right|$。

下面是挑战者 $C$ 和敌手 $A$ 演示的身份基分级密钥封装机制的可选则 ID 的 CPA 安全实验 $G_{A,HIBKEM}^{IND-sID-CPA}(n)$。

a) $A$ 输入它想要挑战的目标身份 $id^*$。

b) 挑战者生成系统参数和最大分级深度并计算 $(MPK, MSK) \xleftarrow{\$} HIBKEM.KGen(1^n, 1^d)$。$C$ 生成 $(K_0, CT^*) \xleftarrow{\$} HIBKEM.Encap(MPK, id^*)$ 和 $K_1 \xleftarrow{\$} \mathrm{K}$。然后挑战者给 $A$ 发送 $(K_b, CT^*, MPK)$，其中 $b \xleftarrow{\$} \{0,1\}$。

c) $A$ 可以查询 $HIBKEM.Del$ 预言机。$HIBKEM.Del$ 预言机输出一个请求身份 $id$ 的私钥。唯一的限制是不允许敌手 $A$ 向 $HIBKEM.Del$ 预言机查询 $id^*$ 或者其祖先的私钥。

d) 最后，$A$ 输出一个猜测 $b'$。定义事件 $b = b'$ 表示 $G_{A,HIBKEM}^{IND-sID-CPA}(n) = 1$。

如果 $Adv_{A,HIBKEM}^{IND-sID-CPA}(n)$ 对所有概率多项式敌手 $A$ 在安全参数 $n$ 上是一个可忽略的函数，那么一个身份基分级密钥封装机制 HIBKEM 是可选则 ID CPA 安全(IND-sID-CPA)。

### 2.3　可穿透前向保密密钥封装

一个可穿透前向保密密钥封装机制 PFSKEM 由五个概率多项式时间算法组成(PFSKEM.KGen，PFSKEM.Encap，PFSKEM.PnctCxt，PFSKEM.Decap，PFSKEM. PnctInt)。

PFSKEM.KGen($1^n$)：算法输入安全参数 $n$ 生成 $(MPK, MSK) \xleftarrow{\$} HIBKEM.KGen(1^n, 1^d)$ 并输出 $PK := MPK$ 和 $SK := (MSK, \varepsilon)$。

PFSKEM.Encap($PK, \tau$)：算法输入公钥和时间间隔 $\tau$，生成 $(pk_{OT}, sk_{OT}) \xleftarrow{\$} OTSIG.KGen(1^n)$。然后计算 $(CT_{HIBKEM}, K) \xleftarrow{\$} HIBKEM.Encap(MPK, \tau \| pk_{OT})$ 和 $\sigma \xleftarrow{\$} OTSIG(sk_{OT}, CT_{HIBKEM})$。让 $CT_{PFSKEM} = (CT_{HIBKEM}, \sigma, pk_{OT})$，输出 $K$ 和 $CT_{PFSKEM}$。

PFSKEM.PnctCxt($SK, \tau, CT_{PFSKEM}$)：把 $CT_{PFSKEM}$ 解析为 $(CT_{HIBKEM}, \sigma, pk_{OT})$，让 $T$ 表示 HIBKEM 树。计算 $SK' = PunctureTree(T, SK, \tau \| pk_{OT})$，输出新密钥 $SK'$。

PFSKEM.Decap($SK, \tau, CT_{PFSKEM}$)解密算法：把 $CT_{PFSKEM}$ 解析为 $(CT_{HIBKEM}, \sigma, pk_{OT})$。如果 $OTSIG.Vfy(pk_{OT}, CT_{HIBKEM}, \sigma) = 0$ 输出 $\perp$。否则：

a) 如果 $SK$ 包含 $id = \tau \| pk_{OT}$ 的私钥 $sk_{id}$，就输出 $K \xleftarrow{\$} HIBKEM.Decap(MPK, sk_{id}, CT_{HIBKEM})$。

b) 如果 $SK$ 包含拥有标签 $id = \tau \| pk_{OT}$ 的节点的一个祖先节点 $n_j$，就计算 $sk_{id} \xleftarrow{\$} HIBKEM.Del(MPK, sk_j, id)$，输出 $K \xleftarrow{\$} HIBKEM.Decap(MPK, sk_{id}, CT_{HIBKEM})$。

c) 否则输出 $\perp$。

PFSKEM.PnctInt($SK, \tau$)下次时间间隔密钥更新算法：计算 $SK' = PunctureTree(T, SK, \tau)$ 其中 $T$ 是 HIBKEM 树。输出用于下次时间间隔 $\tau+1$ 的新密钥 $SK'$。

chinaXiv:202009.00074v1

ChinaXiv合作期刊



PFSKEM 正确性：对所有的 $n \in \mathrm{N}$ ，对任何 $(PK,SK)\overset{\$}{\leftarrow}PFSKEM.KGen(1^n)$ ，时间间隔 $\tau^*$ ， $(K,CT^*)\overset{\$}{\leftarrow}PFSKEM.Encap(PK,\tau^*)$ 和任意交叉序列 $i=0,\ldots,n-1$ ，对任何 $(\tau,CT)\neq(\tau^*,CT^*)$ ，调用 $SK'\overset{\$}{\leftarrow}PFSKEM.PnctCxt(SK,\tau,CT)$ ，或者对任何 $\tau\neq\tau^*$ ，调用 $SK'\overset{\$}{\leftarrow}PFSKEM.PnctInt(SK,\tau)$ ，可以得到 $PFSKEM.Decap(SK',\tau^*, CT^*)=K$ 。

PFSKEM 安全性：在选择时间 CCA 游戏 $G_{A,PFSKEM}^{IND-sT-CCA}(n)$ 中，定义一个敌手 $A$ 的优势为 $Adv_{A,PFSKEM}^{IND-sT-CCA}(n):=\left|\Pr[G_{A,PFSKEM}^{IND-sT-CCA}(n)=1]-1/2\right|$ 。

一个 PFSKEM 方案安全可以通过一个挑战者 $C$ 和一个攻击者 $A$ 演示的选择时间 CCA 安全实验 $G_{A,PFSKEM}^{IND-sT-CCA}(n)$ 来定义。

a) 开始 $A$ 输出目标时间 $\tau^*$ 。

b) 挑战者 $C$ 生成一个新鲜密钥对 $(PK,SK)\overset{\$}{\leftarrow}PFSKEM.KGen(1^n)$ 。计算 $(CT^*,K_0^*)\overset{\$}{\leftarrow}PFSKEM.Encap(PK,\tau^*)$ 并选择 $K_1^*\overset{\$}{\leftarrow}K$ 。此外选择一个比特 $b\overset{\$}{\leftarrow}\{0,1\}$ 然后发送 $(PK,CT^*,K_b^*)$ 给 $A$。

c) $A$ 现在可以进行多项式次下面的查询：

(a) PFSKEM.Decap( $\tau,CT$ )：挑战者计算 $K\overset{\$}{\leftarrow}PFSKEM.Decap(SK,\tau,CT)$ ，返回 $K$ 给 $A$。

(b) PFSKEM.PnctCxt( $\tau,CT$ )：挑战者运行 $SK\overset{\$}{\leftarrow}PFSKEM.PnctCxt(SK,\tau,CT)$ 并返回符号 T。

(c) PFSKEM.PnctInt( $\tau$ )：挑战者运行 $SK\overset{\$}{\leftarrow}PFSKEM.PnctInt(SK,\tau)$ 并返回符号 T。

(d) PFSKEM.Corrupt()：挑战者中止游戏输出一个随机比特如果 $A$ 之前没有查询过 $PFSKEM.PnctCxt(\tau^*,CT^*)$ 或者 $PFSKEM.PnctInt(\tau^*)$ 。否则挑战者返回当下私钥 $SK$ 给 $A$。

d) $A$ 最终输出一个猜测 $b'$ 。用 $G_{A,PFSKEM}^{IND-sT-CCA}(n)$ 表示事件 $b=b'$ 。

如果 $Adv_{A,PFSKEM}^{IND-sT-CCA}(n)$ 在安全参数 $n$ 下对所有概率多项式时间敌手 $A$ 是一个可忽略的函数，那么一个可穿透前向保密密钥封装机制 PFSKEM 是选择时间 CCA 安全(IND-sT-CCA)。

### 2.4　协议描述

本文从可穿透的前向安全密钥封装机制出发构造了一种格上具有前向安全的 0-RTT 密钥交换协议。协议假设客户端和服务器拥有一些大致同步的时间，但强调协议关注的是时间间隔而不是确切的时间，如果时间间隔为一天，则对于本文的方案是有效的。

本文仅在客户端侧针对服务器进行单向认证，客户端不能持有长期密钥材料(即客户端 $pk=\perp$ )，仅能使用其密钥存储当前时间间隔。协议使用穿透加密更新机制在时间间隔结束后或者调用密钥更新算法时更新客户端和服务器的公钥和私钥，同时客户端调用由一次性签名和分级身份基密钥封装机制构造得到的可穿透前向保密密钥封装算法不仅可以生成安全的会话密钥和仅能解密一次的密文，还可以实现认证服务器身份的功能，保证了客户端的安全性。协议具体流程如下：

a) 调用算法 FSOPKE.KGen 为服务器生成带有身份信息的公钥 $pk\leftarrow(PK,\tau_{\max})$ 和私钥 $sk\leftarrow(SK,\tau,\tau_{\max})$ ，为客户端生成公钥 $pk\leftarrow\perp$ 和私钥 $sk\leftarrow(\tau)$ 。

b) 客户端调用算法 FSOPKE.RunC，输入客户端的私钥和服务器的公钥，判断条件 $\tau>\tau_{\max}$ 是否成立，如果条件成立则中止协议并输出 $(sk,\perp,\perp)$ ，否则调用可穿透前向保密密钥封装算法 $PFSKEM.Encap$ ，算法 $PFSKEM.Encap$ 生成一次性签名算法的公钥 $pk_{OT}$ 和私钥 $sk_{OT}$ ，然后调用分级身份基密钥封装算法 $HIBKEM.Encap$ 生成会话密钥 $K$ 和密文 $CT_{HIBKEM}$ ，对 $CT_{HIBKEM}$ 签名得到 $\sigma$ ，算法 FSOPKE.RunC 输出会话密钥 $k\leftarrow K$ 和密文 $m\leftarrow CT_{PFSKEM}=(CT_{HIBKEM},\sigma,pk_{OT})$ ，客户端把密文 $m$ 发送给服务器。

c) 服务器收到密文 $m$ 后调用算法 FSOPKE.RunS，输入服务器的私钥和客户端的公钥以及密文 $m$，判断条件 $SK=\perp$ 或 $\tau>\tau_{\max}$ 是否成立，条件成立输出 $(sk,\perp)$ ，否则调用可穿透前向保密密钥解封装算法 $PFSKEM.Decap$ ，算法 $PFSKEM.Decap$ 调用一次性签名验签算法 $OTSIG.Vfy$ 验证签名，签名成立则调用分级身份基密钥解封装算法 $HIBKEM.Decap$ 解封装得到会话密钥 $K$。

**定义 8**　(FSOPKE, Forward-Secret One-Pass Key Exchange) 一个支持 $\tau_{\max}$ 时间周期并提供单方(仅服务器)身份验证的前向保密一次通过密钥交换(FSOPKE)协议由以下四种概率算法组成。

FSOPKE.KGen $(1^n,r,\tau_{\max})\rightarrow(pk,sk)$ 密钥生成算法：输入安全参数 $n$,一个角色 $r\in\{client,server\}$ 和最大时间周期 $\tau_{\max}\in N$ ，如果 $r$=$server$，生成一个公私钥对 $(PK,SK)\leftarrow PFSKEM.KGen(1^n)$ 。让 $pk\leftarrow(PK,\tau_{\max})$ ， $\tau\leftarrow1$ 和 $sk\leftarrow(SK,\tau,\tau_{\max})$ ，输出 $(pk,sk)$ 。如果 $r$=$client$，让 $pk\leftarrow\perp$ ， $\tau\leftarrow1$ 和 $sk\leftarrow(\tau)$ ，输出 $(pk,sk)$ 。

FSOPKE.RunC $(sk,pk)\rightarrow(sk',k,m)$ 客户端生成会话密钥算法：输入客户端的私钥和服务器的公钥，解析 $sk=(\tau)$ 和 $pk=(PK,\tau_{\max})$ 。如果 $\tau>\tau_{\max}$ ，那么中止协议并输出 $(sk,\perp,\perp)$ ，否则计算 $(CT,K)\leftarrow PFSKEM.Encap(PK,\tau)$ ，让 $k\leftarrow K$ 和 $m\leftarrow CT$ ，输出 $(sk,k,m)$ 。

FSOPKE.RunS $(sk,pk=\perp,m)\rightarrow(sk',k)$ 服务器生成会话密钥算法：输入服务器的私钥、客户端的公钥和密文消息 $m$，解析 $sk=(SK,\tau,\tau_{\max})$ 。如果 $SK=\perp$ 或 $\tau>\tau_{\max}$ ，中止协议并输出 $(sk,\perp)$ 。计算 $K\leftarrow PFSKEM.Decap(SK,\tau,m)$ ，如果 $K=\perp$ ，中止协议并输出 $(sk,\perp)$ ，否则计算 $SK'\leftarrow PFSKEM.PnctCxt(SK,\tau,m)$ 。让 $sk\leftarrow(SK',\tau,\tau_{\max})$ 和 $k\leftarrow K$ ，输出 $(sk,k)$ 。

FSOPKE.TimeStep $(sk,r)\rightarrow sk'$ 时间间隔密钥更新算法：如果 $r$=$server$ 解析 $sk\leftarrow(SK,\tau,\tau_{\max})$ 。如果 $\tau\geq\tau_{\max}$ 让 $sk\leftarrow(\perp,\tau+1,\tau_{\max})$ ，输出 $sk$ 。否则让 $SK'\leftarrow PFSKEM.PnctInt(SK,\tau)$ ， $sk\leftarrow(SK',\tau+1,\tau_{\max})$ ，输出 $sk$ 。如果 $r$=$client$ 解析 $sk=(\tau)$ ，让 $sk\leftarrow(\tau+1)$ 输出 $sk$ 。

0-RTT 密钥交换协议正确性：协议正确性可由底层的 PFSKEM 正确性得到。

### 2.5　安全性证明

本节给出所设计的格上具有前向安全的 0-RTT 密钥交换协议的安全性证明。

**定理 2**　定义 8 中的 FSOPKE 构造是一个安全的单边认证 FSOPKE 协议，在 FSOPKE-sec 游戏中，对于任何有效的对手 $A$，都存在一个有效的算法 B，其算法概率公式为: $Adv_{A,FSOPKE}^{FSOPKE-\sec}(n)\leq n_I\cdot\hat{\tau}_{\max}\cdot n_s\cdot Adv_{B,PFSKEM}^{IND-sT-CCA}(n)$ ，公式中 $n_I=|I|$ 是最大标识数，$\hat{\tau}_{\max}$ 是任何会话的最大时间间隔，$n_s$ 是最大会话数。

**证明**　让 $A$ 成为一个破坏 FSOPKE 安全的敌手并进行一系列的游戏，将引入的差异限定在 $A$ 的每一步的优势中。

游戏 $G_0$ ：这是最初的安全实验，敌手的优势 $Adv_0=Adv_{A,FSOPKE}^{FSOPKE-\sec}(n)$ 。

游戏 $G_1$ ：在这里，让挑战者预先猜测一个与公钥/私钥对 $(pk^*,sk^*)$ 相关联的服务器身份 $s^*\in I$ ，如果这不是测试会话中所涉及的身份，则中止游戏。即如果测试了服务器会话( $\pi^t.role=server$ )，挑战者的猜测是会话所有者 $s^*=\pi^t.id$ ，而如果测试了客户端会话( $\pi^t.role=client$ )，则挑战者的猜测是预期的合作伙伴( $s^*=\pi^t.pid$ )。让 $n_I=|I|$ ，然后 $\mathrm{Adv}_0\leq n_I\cdot\mathrm{Adv}_1$ 。

游戏 $G_2$ ：现在 $A$ 猜测时间间隔 $\tau^*=\pi^t.time$ ，其中测试会话在运行中，如果猜测不对中止游戏。让 $\hat{\tau}_{\max}$ 表示任何会话 $\pi$ 的 $\pi.time$ 最大值，它遵循 $\mathrm{Adv}_1\leq\hat{\tau}_{\max}\cdot\mathrm{Adv}_2$ 。

游戏 $G_3$ ：从 $G_2$ 开始，挑战者如果没有正确地猜测所涉及的客户端会话 $\pi_c^t$ (即， $\pi_c^t.role=client$ )，则遵循下面两个条件的一个：

a) $\pi_c^t=\pi^t$ ，即 $\pi_c^t$ 是被测试的会话，或者

b) $\pi_c^t$ 与测试的(服务器)会话 $\pi^t$ 合作。

对于条件 b)，如果测试服务器会话，必须存在这样的合作客户端会话 $\pi_c^t$ ，其具有 $\pi_c^t.pid=\pi^t.id$ 以便 $A$ 获胜。

chinaXiv:202009.00074v1

ChinaXiv合作期刊



用 $n_s$ 表示为会话总数，有 $Adv_2 \le n_s \cdot Adv_3$ 。

此外，如果测试服务器会话，则会话 $\pi^t$ 实际上必须是 $s^*$ 拥有的第一个接受会话，该会话与 $\pi_c^t$ 合作以便 $A$ 获胜。 回想一下，通过正确性，第一个这样的接受会话派生密钥 $K \neq \perp$ 为 $K \leftarrow PFSKEM.Decap(SK^*, \tau^*, m)$ (其中 $m = \pi^t.trans$ )因此调用 $SK^* \leftarrow PFSKEM.PnctCxt(SK^*, \tau^*, m)$ 。任何后来这样的接受会话将因此派生出 $K = \perp$ ，通过 $K \leftarrow PFSKEM.Decap(SK^*, \tau^*, m)$ 得到，因此敌手将被给予 $\perp$ 作为对其 $Test$ 查询的响应而无法获胜。

游戏 $G_4$：在这个游戏中，将在测试会话 $\pi^t$ 中派生的密钥 $k^*$ 替换为从 PFSKEM.Decap 的输出空间中随机均匀选择的一个。然而，任何能区分从 $G_3$ 到 $G_4$ 的优势不可忽略的对手都可以变成一种算法 B，它以同样的优势在 $G_{A,PFSKEM}^{IND-sT-CCA}$ 中获胜。

在这个归约中，B 首先输出 $G_2$ 中猜测的时间间隔 $\tau^*$ 作为它在 $G_{A,PFSKEM}^{IND-sT-CCA}$ 想要挑战的时间间隔。然后，它得到一个挑战公钥 $PK^*$，该公钥 $PK^*$ 与在 $G_1$ 中猜测的拥有 $pk^* = (PK^*, \tau_{max})$ 的服务器身份 $s^*$ 有关联。对于所有其他身份 $u \in I \setminus \{s^*\}$ ，算法 B 根据 FSOPKE.KGen 生成适当的公钥/私钥对。特别是，它为所有其他服务器身份 $s \in S \setminus \{s^*\}$ 生成 PFSKEM 密钥。此外，B 获得一个挑战密文 $CT^*$ 和密钥 $K^*$，其中 $K^*$ 表示封装在 $CT^*$ 中的真实密钥或者独立选择的随机密钥。

让算法 B 用这种方式正确地为 $A$ 模拟安全游戏，如果 $K^*$ 是真正的密钥，它将完美地模拟 $G_3$，而如果 $K^*$ 是随机选择的密钥，它将完美地模拟 $G_4$。在这个程度上，算法 B 会使用在 PFSKEM 安全性中给出的选择 ID,选择时间 CCA 安全游戏中的预言 PFSKEM.KGen()，PFSKEM.Decap()，PFSKEM.PnctInt()和 PFSKEM.PnctCxt()，在密钥交换游戏对 $A$ 的查询回答如下所示。

$NewSession(u, role, pid, m)$ ，该查询方法需要区分以下情况：

a) 对于所有客户端会话 $\pi_u^t$ ( $u \in C$ )，除了在 $G_3$ 中猜测的客户端会话 $\pi_c^t$ 外，B 模拟安全游戏中指定的 $NewSession$ 查询。

b) 对于猜测的客户端会话 $\pi_c^t$，B 不调用 PFSKEM.Encap，而是使用其挑战密钥 $K^*$ 作为会话密钥 $k$，使用挑战密文 $CT^*$ 作为输出消息 $m$。通过 $G_1$ 到 $G_3$，确保 $\pi_c^t$ 使用服务器 $s^*$ 的时间间隔 $\tau^*$ 和公钥 $pk^*$ 。

c) 对于 $G_1$ 中所有不属于猜测服务器身份 $s^*$ (即 $s \in S \setminus \{s^*\}$ ) 的服务器会话 $\pi_s^t$，B 使用相应的(自己生成的)密钥 $sk_s$ 模拟指定的 $NewSession$ 查询。

d) 对于所有属于 $s^*$ 的服务器会话 $\pi_{s^*}^i$，但未与猜测的客户端会话 $\pi_c^t$ 合作，B 根据预言 PFSKEM.Decap 和 PFSKEM.PnctCxt 来模拟 $NewSession$ 查询的操作。因为 $\pi_{s^*}^i$ 和 $\pi_c^t$ 不是合作会话(尽管拥有相反的角色且 $\pi_c^t.pid = s^*$ )，且 $(\pi_c^t.time, \pi_c^t.trans) = (\tau^*, CT^*) \neq (\pi_{s^*}^i.time, \pi_{s^*}^i.trans)$ ，所以允许调用 PFSKEM.Decap 预言作为此处的输入。

e) 对于 $s^*$ 拥有的第一个服务器会话 $\pi_{s^*}^t$，其与猜测的客户端会话 $\pi_c^t$ 合作，B 将会话密钥设置为挑战密钥 $k \leftarrow K^*$ 并调用 $PFSKEM.PnctCxt(\tau^*, CT^*)$ 。合作意味着 $\pi_{s^*}^t$ 与 $\pi_c^t$ 保持相同的时间，并获得 $\pi_c^t$ 的消息，即 $\pi_c^t.time = \tau^* = \pi_{s^*}^t.time$ 和 $\pi_c^t.trans = m = \pi_{s^*}^t.trans$ 。此外，PFSKEM.PnctCxt 不会在 $(\tau^*, CT^*)$ 之前被调用。由正确性可以得到 $\pi_{s^*}^t$ 和 $\pi_c^t$ 建立了相同的会话密钥 $K^*$ 。

f) 对于与 $\pi_c^t$ 合作的任何其他服务器会话 $\pi_{s^*}^i$， B 设置 $K \leftarrow \perp$ 。任何此类会话都将得到 $\perp \leftarrow PFSKEM.Decap(SK, \tau^*, CT^*)$ ，因为 PFSKEM.PnctCxt 已经在 $(\tau^*, CT^*)$ 之前被调用。

$Reveal(\pi_u^t)$：首先，观察任何获胜的敌手 $A$ 不能根据匹配会话的条件 a)b)在会话 $\pi_c^t$ 和 $\pi_{s^*}^t$ 上公开，因为其中一个是测试会话，如果存在另一个，那么它会与测试会话合作。

对于所有其他会话，B 持有上述 $NewSession$ 查询中模拟得到的正确密钥，因此可以根据指定的 $Reveal$ 查询进行响应。

$Corrupt(u)$：对于测试会话 $\pi^t$ 中涉及的服务器身份 $s^*$，B 调用其 $PFSKEM.Corrupt$ 预言获取 PFSKEM 密钥 $SK^*$，该密钥在 $sk^* = (SK^*, \tau_{s^*}, \tau_{max})$ 内返回。如果 $A$ 无丢失调用 $Corrupt(s^*)$，确保 B 在 $Corrupt(s^*)$ 之前调用了 $PFSKEM.PnctCxt(\tau^*, CT^*)$ 或者 $PFSKEM.PnctInt(\tau^*)$，因此在选择性时间 CCA 安全游戏中不会丢失：

如果 $\pi^t = \pi_{s^*}^t$ 是一个服务器会话(由 $s^*$ 拥有)，匹配会话的条件 c 确保 $s^*$ 只有在 $\pi^t$ 被接受后才能被中断。在 $\pi^t$ 接受过程中存在 $\pi^t.time = \tau^*$ 和 $\pi^t.trans = CT^*$，因此在 $s^*$ 中断之前，B 必须调用 $PFSKEM.PnctCxt(\tau^*, CT^*)$ 。如果 $\pi^t = \pi_c^t$ 是一个客户端会话，那么匹配会话的条件 d 确保存在一个在时间间隔 $\tau^*$ 中处理 $CT^*$ 的合作服务器会话 $\pi_{s^*}^t$，或者 $s^*$ 在时间间隔 $\tau_{s^*}^{corr} > \pi^t.time = \tau^*$ 中被中断。因此，B 必须在 $s^*$ 被中断之前分别调用 $PFSKEM.PnctCxt(\tau^*, CT^*)$ 和 $PFSKEM.PnctInt(\tau^*)$ 。

对于任何其他(客户端或服务器)身份 $u \neq s^*$，B 维护相应的私钥 $sk_u$，因此可以响应指定的 $Corrupt$ 查询。

$Tick(u)$：算法 B 使用其对于未知私钥 $SK^*$ 的预言 PFSKEM.PnctInt 执行指定的时间步进过程，该私钥与 PFSKEM 挑战公钥 $PK^*$ 有关。

$Test(\pi^t)$：测试会话 $\pi^t$ 必须是 $G_3$ 中猜测的客户端会话 $\pi_c^t$，或者是属于与 $\pi_c^t$ 合作的 $s^*$ 的第一个服务器会话 $\pi_{s^*}^t$， 在这两种情况下，算法 B 只输出 $\pi^t.key = K^*$ 作为 $Test$ 查询的响应。

当 $A$ 停止并输出一个猜测 $b \in \{0,1\}$，B 也停止把 $b$ 作为自己的猜测输出。

算法 B 正确回答 $A$ 的所有查询，在 $K^*$ 是封装在 $CT^*$ 中的真正密钥的情况下，则它完全模拟 $G_3$，而如果 $K^*$ 是随机独立选择的，则它完全模拟 $G_4$。而且如果 $A$ 坚持 FSOPKE 安全游戏的条件，那么算法 B 就会遵循在 PFSKEM 安全性中选择 ID 选择时间 CCA 安全游戏的所有限制。

当 B 继承 $A$ 的输出时，$A$ 在 $G_3$ 中的优势与其在 $G_4$ 中的优势之间的差异与 B 在选择 ID、选择时间 CCA 安全实验两种情况下输出的概率差异有关。因此，

$$Adv_3 \le Adv_4 + Adv_{B,PFSKEM}^{IND-sT-CCA}(n) \text{ 。}$$

在 $G_4$ 中，测试会话中的会话密钥 $k^*$ 总是随机均匀选择，对查询 $Test$ 的响应与挑战位 $b$ 无关，因此 $A$ 不能比猜测更好地预测 $b$，即 $Adv_4 \le 0$ 。结合 $G_1$ 到 $G_4$ 中的优势边界，得出整体边界。

## 3 性能分析

本节从安全性和效率两方面，对本文协议和 Katz 等[15]提出的 PAKE 协议和 Zhang 等[18]提出的 PAKE 协议进行比较，这些协议都是由格上困难问题构造得到。3 中协议性能如表 1 所示。

在安全性方面，协议引入了穿透加密机制，提供了完全前向安全性，与协议[15][18]相比，本文协议可以抵抗重放攻击，具有更高的安全性。在效率方面，协议由一次性签名技术和分级身份基密钥封装构造得到，完成密钥协商只需要 1 轮通信。由表 1 可以看出，与协议[15][18]相比，本文协议的通信开销较小，主要由分级身份基密钥封装的密文及其签名构成，这使本文协议的通信效率提高。

表 1 三种方案性能比较

Tab. 1 Performance comparison of three schemes

| 方案 | 类型 | 完全前向安全 | 通信轮数 | 通信开销 | 服务器计算开销 | 用户计算开销 |
|---|---|---|---|---|---|---|
| Katz 等协议 | 2-party | 否 | 3 | $2m+4n$ | $O(mn)$ | $O(mn)$ |
| Zhang 等协议 | 2-party | 否 | 2 | $4m+2n_1+l$ | $O(mn)$ | $O(mn)$ |
| 本文协议 | 2-party | 是 | 1 | $2m+1$ | $O(nm^3)$ | $O(nm^3)$ |

chinaXiv:202009.00074v1

ChinaXiv合作期刊


协议[15]是基于 LWE 的 2PAKE 协议，需要 3 轮通信，通信代价由密文、投射密钥和消息认证码决定，大小为 $2m$，通信开销比本文多 $4n-1$。此外，本文使用穿透加密提供了完全前向安全性。根据分析和表 1 数据可得，本文协议不仅具有更强的前向安全性，而且通信效率更高。

协议[18]是基于格的 2PAKE 协议，需要 2 轮通信，通信代价主要取决于密文和投射密钥的大小，本文协议和协议[18]相比，通信开销只有 $2m+1$，增加了服务器计算开销和用户计算开销。同时本文协议仅需 1 轮通信，提供了更强的前向安全性。根据分析和表 1 数据可得，本文协议通信效率更低，安全性更高。

以上分析结果表明，本文构造的密钥交换协议通过密钥穿透更新策略实现了完全前向安全性，可以抵抗重放攻击、量子攻击；而且本文协议在协商密钥时只需要给服务器发送一条包含加密保护的有效载荷和一条密钥交换协议消息而不需要服务器回复从而实现了 0 轮往返时长通信，极大的降低了通信开销，因此本文协议具有可行性。

## 4 结束语

本文提出的 0-RTT 密钥交换协议，其中一次性签名技术基于格上的 SIS 困难问题，身份基分级密钥封装机制基于格上的 LWE 困难问题，在后量子时代具有重要意义。设计的 0-RTT 密钥交换协议仅需要服务器认证，把会话密钥和传递的消息一起发送给服务器从而实现了 0 轮通信的效果。协议可以有效抵抗量子攻击和重放攻击，提高了应用安全性，同时给出了在标准模型下严格的安全性证明。协议因为安全性和部署参数在执行穿透操作更新密钥时可能要花费较多的时间，但可以通过进行少量的有效计算和删除二叉树中的部分私钥来优化穿透操作降低耗时。

## 参考文献：


[1] Diffie W, Hellman M. New directions in cryptography [J]. IEEE Trans on Information Theory, 1976, 22 (6): 644-654.

[2] Shor, Peter W. Polynomial-Time Algorithms for Prime Factorization and Discrete Logarithms on a Quantum Computer [J]. SIAM Journal on Computing, 1997, 26 (5): 1484-1509.

[3] Hu Zhao, Zhu Yuesheng, Ma Limin. An improved Kerberos protocol based on Diffie-Hellman-DSA key exchange [C]// IEEE International Conference on Networks. Piscataway, NJ: IEEE Press, 2012: 400-404.

[4] Hu Xuexian, Liu Wenfen, Zhang Jianhui. An Efficient ID-Based Authenticated Key Exchange Protocol [C]// Wase International Conference on Information Engineering. Piscataway, NJ: IEEE Press, 2009: 229-233.

[5] Dierks T, Rescorla E. The Transport Layer Security (TLS) Protocol Version 1. 2, RFC 5246 [EB/OL]. [2008-08]. https://www. rfc-editor. org/info/rfc5246.

[6] Rescorla E. The Transport Layer Security (TLS) Protocol Version 1. 3 [EB/OL]. [2016-10]. https://tools. ietf. org/html/draft-ietf-tls-tls13-18.

[7] Krawczyk H. HMQV: A high-performance secure Diffie-Hellman protocol [C]// Advances in Cryptology – CRYPTO 2005. CRYPTO 2005. Lecture Notes in Computer Science. Berlin: Springer, 2005: 546-566.

[8] QUIC, a multiplexed stream transport over UDP [EB/OL]. https://www. helplib. com/c/mutia_121632.

[9] Günther F, Hale B, Jager T, *et al.* 0-RTT Key Exchange with Full Forward Secrecy [C]// International Conference on the Theory & Applications of Cryptographic Techniques. Berlin: Springer, 2017: 519-548.

[10] Canetti R, Halevi S, Katz J. A Forward-Secure Public-Key Encryption Scheme [M]. Advances in Cryptology — EUROCRYPT 2003. Berlin: Springer, 2003: 255-271.

[11] Green M D, Miers I. Forward Secure Asynchronous Messaging from Puncturable Encryption [C]// IEEE Symposium on Security & Privacy. Piscataway, NJ: IEEE Press, 2015: 305-320.

[12] Blazy O, Kiltz E, Pan J. (Hierarchical) Identity-Based Encryption from Affine Message Authentication [M]// Advances in Cryptology – CRYPTO 2014. Berlin: Springer, 2014: 408-425.

[13] Derler D, Jager T, Slamanig D, *et al.* Bloom Filter Encryption and Applications to Efficient Forward-Secret 0-RTT Key Exchange [C]// International Conference on the Theory & Applications of Cryptographic Techniques. Berlin: Springer, 2018: 425-455.

[14] 魏福山, 马建峰, 李光松, 等. 标准模型下高效的三方口令认证密钥交换协议 [J]. 软件学报, 2016, 27 (9): 2389-2399. (Wei Fushan, Ma Jianfeng, Li Guangsong, el al. Efficient Three-Party Password-Based Authenticated Key Exchange Protocol in the Standard Model [J]. Journal of Software, 2016, 27 (9): 2389-2399.)

[15] Katz J, Vaikuntanathan V. Smooth Projective Hashing and Password-Based Authenticated Key Exchange from Lattices [C]// Advances in Cryptology-ASIACRYPT 2009, the 15th International Conference on the Theory and Application of Cryptology and Information Security. Berlin: Springer, 2009: 636-652.

[16] Ding Yi, Fan Lei. Efficient password-based authenticated key exchange from lattices [C]// Seventh International Conference on Computational Intelligence and Security. Piscataway, NJ: IEEE Press, 2012: 934-938.

[17] Groce A, Katz J. A New Framework for Efficient Password-Based Authenticated Key Exchange [C]// Proceedings of the 17th ACM Conference on Computer and Communications Security. New York: ACM Press, 2010: 516-525.

[18] Zhang Jiang, Yu Yu. Two-Round PAKE from Approximate SPH and Instantiations from Lattices [C]// International Conference on the Theory and Application of Cryptology and International Security. Berlin: Springer, 2017: 37-67.

[19] 李子臣, 谢婷, 张卷美, 等. 基于 RLWE 的后量子认证密钥交换协议 [J]. 计算机研究与发展, 2019, 56 (12): 2694-2701. (Li Zichen, Xie Ting, Zhang Juanmei, *et al.* Post Quantum Authenticated Key Exchange Protocol Based on Ring Learning with Errors Problem [J]. Journal of Computer Research and Development, 2019, 56 (12): 2694-2701.)

[20] Bellare M, Rogaway P. Entity Authentication and Key Distribution [C]// Annual International Conference on the Theory and Applications of Cryptographic Techniques. Berlin: Springer, 1993: 232-249.

[21] Micciancio D, Peikert C. Trapdoors for lattices: simpler, tighter, faster, smaller [C]. Advances in Cryptology – EUROCRYPT 2012. EUROCRYPT 2012. Lecture Notes in Computer Science. Berlin: Springer, 2012: 700 – 718.


chinaXiv:202009.00074v1